

# Hosting Environment (Daemon) Chain Components Reference Manual

Generated by Doxygen 1.4.7

Mon Apr 23 10:06:43 2012



# Contents

<b>1</b>	<b>Hosting Environment (Daemon) Chain Components Namespace Index</b>	<b>1</b>
1.1	Hosting Environment (Daemon) Chain Components Namespace List . . . . .	1
<b>2</b>	<b>Hosting Environment (Daemon) Chain Components Hierarchical Index</b>	<b>3</b>
2.1	Hosting Environment (Daemon) Chain Components Class Hierarchy . . . . .	3
<b>3</b>	<b>Hosting Environment (Daemon) Chain Components Data Structure Index</b>	<b>5</b>
3.1	Hosting Environment (Daemon) Chain Components Data Structures . . . . .	5
<b>4</b>	<b>Hosting Environment (Daemon) Chain Components Namespace Documentation</b>	<b>7</b>
4.1	ArcSec Namespace Reference . . . . .	7
<b>5</b>	<b>Hosting Environment (Daemon) Chain Components Data Structure Documentation</b>	<b>11</b>
5.1	ArcSec::AllowPDP Class Reference . . . . .	11
5.2	ArcSec::ArcAlgFactory Class Reference . . . . .	12
5.3	ArcSec::ArcAttributeFactory Class Reference . . . . .	13
5.4	ArcSec::ArcAttributeProxy< TheAttribute > Class Template Reference . . . . .	14
5.5	ArcSec::ArcAuthZ Class Reference . . . . .	15
5.6	ArcSec::ArcEvaluationCtx Class Reference . . . . .	16
5.7	ArcSec::ArcEvaluator Class Reference . . . . .	17
5.8	ArcSec::ArcFnFactory Class Reference . . . . .	18
5.9	ArcSec::ArcPDP Class Reference . . . . .	19
5.10	ArcSec::ArcPolicy Class Reference . . . . .	20
5.11	ArcSec::ArcRequestItem Class Reference . . . . .	21
5.12	ArcSec::ArcRequestTuple Class Reference . . . . .	22
5.13	ArcSec::ArcRule Class Reference . . . . .	23
5.14	Arc::DataPointARC Class Reference . . . . .	24
5.15	Arc::DataPointFile Class Reference . . . . .	25
5.16	Arc::DataPointGFAL Class Reference . . . . .	26
5.17	Arc::DataPointGridFTP Class Reference . . . . .	27

5.18	Arc::DataPointHTTP Class Reference . . . . .	28
5.19	Arc::DataPointLDAP Class Reference . . . . .	29
5.20	Arc::DataPointLFC Class Reference . . . . .	30
5.21	Arc::DataPointRLS Class Reference . . . . .	31
5.22	Arc::DataPointSRM Class Reference . . . . .	32
5.23	Arc::DataPointXrootd Class Reference . . . . .	33
5.24	ArcSec::DelegationPDP Class Reference . . . . .	34
5.25	ArcSec::DenyPDP Class Reference . . . . .	35
5.26	Arc::LDAPQuery Class Reference . . . . .	36
5.27	ArcMCCHTTP::MCC_HTTP Class Reference . . . . .	37
5.28	ArcMCCHTTP::MCC_HTTP_Client Class Reference . . . . .	38
5.29	ArcMCCHTTP::MCC_HTTP_Service Class Reference . . . . .	39
5.30	ArcMCCSOAP::MCC_SOAP Class Reference . . . . .	40
5.31	ArcMCCSOAP::MCC_SOAP_Service Class Reference . . . . .	41
5.32	ArcMCCTCP::MCC_TCP Class Reference . . . . .	42
5.33	ArcMCCTCP::MCC_TCP_Client Class Reference . . . . .	43
5.34	ArcMCCTCP::MCC_TCP_Service Class Reference . . . . .	44
5.35	ArcMCCTLS::MCC_TLS Class Reference . . . . .	45
5.36	ArcMCCTLS::MCC_TLS_Client Class Reference . . . . .	46
5.37	ArcMCCTLS::MCC_TLS_Service Class Reference . . . . .	47
5.38	ArcMCCHTTP::PayloadHTTP Class Reference . . . . .	48
5.39	ArcMCCTCP::PayloadTCPsocket Class Reference . . . . .	53
5.40	ArcMCCTLS::PayloadTLSStream Class Reference . . . . .	54
5.41	ArcSec::PDPServiceInvoker Class Reference . . . . .	56
5.42	ArcSec::SAML2SSO_AssertionConsumerSH Class Reference . . . . .	57
5.43	ArcSec::SAMLTokenSH Class Reference . . . . .	58
5.44	ArcSec::SimpleListPDP Class Reference . . . . .	59
5.45	Arc::SRMClient Class Reference . . . . .	60
5.46	Arc::SRMClientRequest Class Reference . . . . .	69
5.47	SRMFileInfo Class Reference . . . . .	71
5.48	Arc::SRMFileMetaData Struct Reference . . . . .	72
5.49	SRMInfo Class Reference . . . . .	73
5.50	Arc::SRMInvalidRequestException Class Reference . . . . .	74
5.51	ArcSec::UsernameTokenSH Class Reference . . . . .	75
5.52	ArcSHCLegacy::voms Struct Reference . . . . .	76
5.53	ArcSHCLegacy::voms_attrs Struct Reference . . . . .	77

5.54 ArcSec::X509TokenSH Class Reference . . . . .	78
5.55 ArcSec::XACMLAlgFactory Class Reference . . . . .	79
5.56 ArcSec::XACMLAttributeFactory Class Reference . . . . .	80
5.57 ArcSec::XACMLAttributeProxy< TheAttribute > Class Template Reference . . . . .	81
5.58 ArcSec::XACMLCondition Class Reference . . . . .	82
5.59 ArcSec::XACMLEvaluationCtx Class Reference . . . . .	83
5.60 ArcSec::XACMLEvaluator Class Reference . . . . .	84
5.61 ArcSec::XACMLFnFactory Class Reference . . . . .	85
5.62 ArcSec::XACMLPDP Class Reference . . . . .	86
5.63 ArcSec::XACMLPolicy Class Reference . . . . .	87
5.64 ArcSec::XACMLRule Class Reference . . . . .	88
5.65 ArcSec::XACMLTarget Class Reference . . . . .	89



# Chapter 1

## Hosting Environment (Daemon) Chain Components Namespace Index

### 1.1 Hosting Environment (Daemon) Chain Components Namespace List

Here is a list of all documented namespaces with brief descriptions:

[ArcSec](#) (ArcRequest, Parsing the specified Arc request format ) . . . . . 7





## Chapter 2

# Hosting Environment (Daemon) Chain Components Hierarchical Index

### 2.1 Hosting Environment (Daemon) Chain Components Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ArcSec::AllowPDP . . . . .	11
ArcSec::ArcAlgFactory . . . . .	12
ArcSec::ArcAttributeFactory . . . . .	13
ArcSec::ArcAttributeProxy< TheAttribute > . . . . .	14
ArcSec::ArcAuthZ . . . . .	15
ArcSec::ArcEvaluationCtx . . . . .	16
ArcSec::ArcEvaluator . . . . .	17
ArcSec::ArcFnFactory . . . . .	18
ArcSec::ArcPDP . . . . .	19
ArcSec::ArcPolicy . . . . .	20
ArcSec::ArcRequestItem . . . . .	21
ArcSec::ArcRequestTuple . . . . .	22
ArcSec::ArcRule . . . . .	23
Arc::DataPointARC . . . . .	24
Arc::DataPointFile . . . . .	25
Arc::DataPointGFAL . . . . .	26
Arc::DataPointGridFTP . . . . .	27
Arc::DataPointHTTP . . . . .	28
Arc::DataPointLDAP . . . . .	29
Arc::DataPointLFC . . . . .	30
Arc::DataPointRLS . . . . .	31
Arc::DataPointSRM . . . . .	32
Arc::DataPointXrootd . . . . .	33
ArcSec::DelegationPDP . . . . .	34
ArcSec::DenyPDP . . . . .	35
Arc::LDAPQuery . . . . .	36
ArcMCCHTTP::MCC_HTTP . . . . .	37
ArcMCCHTTP::MCC_HTTP_Client . . . . .	38
ArcMCCHTTP::MCC_HTTP_Service . . . . .	39

ArcMCCSOAP::MCC_SOAP . . . . .	40
ArcMCCSOAP::MCC_SOAP_Service . . . . .	41
ArcMCCTCP::MCC_TCP . . . . .	42
ArcMCCTCP::MCC_TCP_Client . . . . .	43
ArcMCCTCP::MCC_TCP_Service . . . . .	44
ArcMCCTLS::MCC_TLS . . . . .	45
ArcMCCTLS::MCC_TLS_Client . . . . .	46
ArcMCCTLS::MCC_TLS_Service . . . . .	47
ArcMCCHTTP::PayloadHTTP . . . . .	48
ArcMCCTCP::PayloadTCPSocket . . . . .	53
ArcMCCTLS::PayloadTLSStream . . . . .	54
ArcSec::PDPServletInvoker . . . . .	56
ArcSec::SAML2SSO_AssertionConsumerSH . . . . .	57
ArcSec::SAMLTokenSH . . . . .	58
ArcSec::SimpleListPDP . . . . .	59
Arc::SRMClient . . . . .	60
Arc::SRMClientRequest . . . . .	69
SRMFileInfo . . . . .	71
Arc::SRMFileMetaData . . . . .	72
SRMInfo . . . . .	73
Arc::SRMInvalidRequestException . . . . .	74
ArcSec::UsernameTokenSH . . . . .	75
ArcSHCLegacy::voms . . . . .	76
ArcSHCLegacy::voms_attrs . . . . .	77
ArcSec::X509TokenSH . . . . .	78
ArcSec::XACMLAlgFactory . . . . .	79
ArcSec::XACMLAttributeFactory . . . . .	80
ArcSec::XACMLAttributeProxy< TheAttribute > . . . . .	81
ArcSec::XACMLCondition . . . . .	82
ArcSec::XACMLEvaluationCtx . . . . .	83
ArcSec::XACMLEvaluator . . . . .	84
ArcSec::XACMLFnFactory . . . . .	85
ArcSec::XACMLPDP . . . . .	86
ArcSec::XACMLPolicy . . . . .	87
ArcSec::XACMLRule . . . . .	88
ArcSec::XACMLTarget . . . . .	89

## Chapter 3

# Hosting Environment (Daemon) Chain Components Data Structure Index

### 3.1 Hosting Environment (Daemon) Chain Components Data Structures

Here are the data structures with brief descriptions:

<a href="#">ArcSec::AllowPDP</a> (This PDP always return true (allow) ) . . . . .	11
<a href="#">ArcSec::ArcAlgFactory</a> (Algorithm factory class for Arc ) . . . . .	12
<a href="#">ArcSec::ArcAttributeFactory</a> (Attribute factory class for Arc specified attributes ) . . . . .	13
<a href="#">ArcSec::ArcAttributeProxy&lt; TheAttribute &gt;</a> (Arc specific AttributeProxy class ) . . . . .	14
<a href="#">ArcSec::ArcAuthZ</a> (Tests message against list of PDPs ) . . . . .	15
<a href="#">ArcSec::ArcEvaluationCtx</a> (EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc ) . . . . .	16
<a href="#">ArcSec::ArcEvaluator</a> (Execute the policy evaluation, based on the request and policy ) . . . . .	17
<a href="#">ArcSec::ArcFnFactory</a> (Function factory class for Arc specified attributes ) . . . . .	18
<a href="#">ArcSec::ArcPDP</a> ( <a href="#">ArcPDP</a> - PDP which can handle the Arc specific request and policy schema ) . . . . .	19
<a href="#">ArcSec::ArcPolicy</a> ( <a href="#">ArcPolicy</a> class to parse and operate Arc specific <Policy> node ) . . . . .	20
<a href="#">ArcSec::ArcRequestItem</a> (Container, <Subjects, Actions, Objects, Contexts> tuple ) . . . . .	21
<a href="#">ArcSec::ArcRequestTuple</a> (RequestTuple, container which includes the ) . . . . .	22
<a href="#">ArcSec::ArcRule</a> ( <a href="#">ArcRule</a> class to parse Arc specific <Rule> node ) . . . . .	23
<a href="#">Arc::DataPointARC</a> . . . . .	24
<a href="#">Arc::DataPointFile</a> . . . . .	25
<a href="#">Arc::DataPointGFAL</a> . . . . .	26
<a href="#">Arc::DataPointGridFTP</a> . . . . .	27
<a href="#">Arc::DataPointHTTP</a> . . . . .	28
<a href="#">Arc::DataPointLDAP</a> . . . . .	29
<a href="#">Arc::DataPointLFC</a> . . . . .	30
<a href="#">Arc::DataPointRLS</a> . . . . .	31
<a href="#">Arc::DataPointSRM</a> . . . . .	32
<a href="#">Arc::DataPointXrootd</a> . . . . .	33
<a href="#">ArcSec::DelegationPDP</a> . . . . .	34
<a href="#">ArcSec::DenyPDP</a> (This PDP always returns false (deny) ) . . . . .	35
<a href="#">Arc::LDAPQuery</a> . . . . .	36
<a href="#">ArcMCCHTTP::MCC_HTTP</a> (A base class for HTTP client and service MCCs ) . . . . .	37
<a href="#">ArcMCCHTTP::MCC_HTTP_Client</a> . . . . .	38

<a href="#">ArcMCCHTTP::MCC_HTTP_Service</a> . . . . .	39
<a href="#">ArcMCCSOAP::MCC_SOAP</a> (A base class for SOAP client and service MCCs ) . . . . .	40
<a href="#">ArcMCCSOAP::MCC_SOAP_Service</a> . . . . .	41
<a href="#">ArcMCCTCP::MCC_TCP</a> (A base class for TCP client and service MCCs ) . . . . .	42
<a href="#">ArcMCCTCP::MCC_TCP_Client</a> . . . . .	43
<a href="#">ArcMCCTCP::MCC_TCP_Service</a> . . . . .	44
<a href="#">ArcMCCTLS::MCC_TLS</a> (A base class for TLS client and service MCCs ) . . . . .	45
<a href="#">ArcMCCTLS::MCC_TLS_Client</a> . . . . .	46
<a href="#">ArcMCCTLS::MCC_TLS_Service</a> . . . . .	47
<a href="#">ArcMCCHTTP::PayloadHTTP</a> . . . . .	48
<a href="#">ArcMCCTCP::PayloadTCPsocket</a> . . . . .	53
<a href="#">ArcMCCTLS::PayloadTLSStream</a> . . . . .	54
<a href="#">ArcSec::PDPSERVICEInvoker</a> ( <a href="#">PDPSERVICEInvoker</a> - client which will invoke pdpservice ) . . . .	56
<a href="#">ArcSec::SAML2SSO_AssertionConsumerSH</a> (Implement the functionality of the Service Provider in SAML2 SSO profile ) . . . . .	57
<a href="#">ArcSec::SAMLTokenSH</a> (Adds WS-Security SAML Token into SOAP Header ) . . . . .	58
<a href="#">ArcSec::SimpleListPDP</a> (Tests X509 subject against list of subjects in file ) . . . . .	59
<a href="#">Arc::SRMClient</a> . . . . .	60
<a href="#">Arc::SRMClientRequest</a> (Class to represent a SRM request ) . . . . .	69
<a href="#">SRMFileInfo</a> . . . . .	71
<a href="#">Arc::SRMFileMetaData</a> (SRM-related file metadata ) . . . . .	72
<a href="#">SRMInfo</a> . . . . .	73
<a href="#">Arc::SRMInvalidRequestException</a> (General exception to represent a bad SRM request ) . . . .	74
<a href="#">ArcSec::UsernameTokenSH</a> (Adds WS-Security Username Token into SOAP Header ) . . . . .	75
<a href="#">ArcSHCLegacy::voms</a> . . . . .	76
<a href="#">ArcSHCLegacy::voms_attrs</a> . . . . .	77
<a href="#">ArcSec::X509TokenSH</a> (Adds WS-Security X509 Token into SOAP Header ) . . . . .	78
<a href="#">ArcSec::XACMLAlgFactory</a> (Algorithm factory class for XACML ) . . . . .	79
<a href="#">ArcSec::XACMLAttributeFactory</a> (Attribute factory class for XACML specified attributes ) . . .	80
<a href="#">ArcSec::XACMLAttributeProxy</a> <a href="#">&lt; TheAttribute &gt;</a> (XACML specific AttributeProxy class ) . . .	81
<a href="#">ArcSec::XACMLCondition</a> ( <a href="#">XACMLCondition</a> class to parse and operate XACML specific <Condition> node ) . . . . .	82
<a href="#">ArcSec::XACMLEvaluationCtx</a> (EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc ) . . . . .	83
<a href="#">ArcSec::XACMLEvaluator</a> (Execute the policy evaluation, based on the request and policy ) . .	84
<a href="#">ArcSec::XACMLFnFactory</a> (Function factory class for XACML specified attributes ) . . . . .	85
<a href="#">ArcSec::XACMLPDP</a> ( <a href="#">XACMLPDP</a> - PDP which can handle the XACML specific request and policy schema ) . . . . .	86
<a href="#">ArcSec::XACMLPolicy</a> ( <a href="#">XACMLPolicy</a> class to parse and operate XACML specific <Policy> node ) . . . . .	87
<a href="#">ArcSec::XACMLRule</a> ( <a href="#">XACMLRule</a> class to parse XACML specific <Rule> node ) . . . . .	88
<a href="#">ArcSec::XACMLTarget</a> ( <a href="#">XACMLTarget</a> class to parse and operate XACML specific <Target> node ) . . . . .	89

## Chapter 4

# Hosting Environment (Daemon) Chain Components Namespace Documentation

### 4.1 ArcSec Namespace Reference

ArcRequest, Parsing the specified Arc request format.

#### Data Structures

- class [AllowPDP](#)  
*This PDP always return true (allow).*
- class [ArcAuthZ](#)  
*Tests message against list of PDPs.*
- class [ArcAlgFactory](#)  
*Algorithm factory class for Arc.*
- class [ArcAttributeFactory](#)  
*Attribute factory class for Arc specified attributes.*
- class [ArcAttributeProxy](#)  
*Arc specific AttributeProxy class.*
- class [ArcRequestTuple](#)  
*RequestTuple, container which includes the.*
- class [ArcEvaluationCtx](#)  
*EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc.*
- class [ArcEvaluator](#)

*Execute the policy evaluation, based on the request and policy.*

- class [ArcFnFactory](#)

*Function factory class for Arc specified attributes.*

- class [ArcPDP](#)

*ArcPDP - PDP which can handle the Arc specific request and policy schema.*

- class [ArcPolicy](#)

*ArcPolicy class to parse and operate Arc specific <Policy> node.*

- class **ArcRequest**

- class [ArcRequestItem](#)

*Container, <Subjects, Actions, Objects, Contexts> tuple.*

- class [ArcRule](#)

*ArcRule class to parse Arc specific <Rule> node.*

- class [DelegationPDP](#)

- class **DelegationSH**

- class [DenyPDP](#)

*This PDP always returns false (deny).*

- class **GACLEvaluator**

- class **GACLPDP**

- class **GACLPolicy**

- class **GACLRequest**

- class [PDPServiceInvoker](#)

*PDPServiceInvoker - client which will invoke pdpservice.*

- class [SAML2SSO\\_AssertionConsumerSH](#)

*Implement the functionality of the Service Provider in SAML2 SSO profile.*

- class [SAMLTokenSH](#)

*Adds WS-Security SAML Token into SOAP Header.*

- class [SimpleListPDP](#)

*Tests X509 subject against list of subjects in file.*

- class [UsernameTokenSH](#)

*Adds WS-Security Username Token into SOAP Header.*

- class [X509TokenSH](#)

*Adds WS-Security X509 Token into SOAP Header.*

- class **AttributeDesignator**

- class **AttributeSelector**

- class [XACMLAlgFactory](#)

*Algorithm factory class for XACML.*

- class **XACMLApply**
- class [XACMLAttributeFactory](#)  
*Attribute factory class for XACML specified attributes.*
- class [XACMLAttributeProxy](#)  
*XACML specific AttributeProxy class.*
- class [XACMLCondition](#)  
*[XACMLCondition](#) class to parse and operate XACML specific <Condition> node.*
- class [XACMLEvaluationCtx](#)  
*EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc.*
- class [XACMLEvaluator](#)  
*Execute the policy evaluation, based on the request and policy.*
- class [XACMLFnFactory](#)  
*Function factory class for XACML specified attributes.*
- class [XACMLPDP](#)  
*[XACMLPDP](#) - PDP which can handle the XACML specific request and policy schema.*
- class [XACMLPolicy](#)  
*[XACMLPolicy](#) class to parse and operate XACML specific <Policy> node.*
- class **XACMLRequest**
- class [XACMLRule](#)  
*[XACMLRule](#) class to parse XACML specific <Rule> node.*
- class **XACMLTargetMatch**
- class **XACMLTargetMatchGroup**
- class **XACMLTargetSection**
- class [XACMLTarget](#)  
*[XACMLTarget](#) class to parse and operate XACML specific <Target> node.*

## Typedefs

- typedef std::pair< AttributeValue \*, Function \* > [Match](#)
- typedef std::list< [Match](#) > [AndList](#)
- typedef std::list< [AndList](#) > [OrList](#)

### 4.1.1 Detailed Description

ArcRequest, Parsing the specified Arc request format.

## 4.1.2 Typedef Documentation

### 4.1.2.1 typedef std::pair<AttributeValue\*, Function\*> [ArcSec::Match](#)

Pair Match include the AttributeValue object in <Rule> and the Function which is used to handle the AttributeValue, default function is "Equal", if some other function is used, it should be explicitly specified, e.g. Subject Type="string" Function="Match"/>/vo.knowarc/usergroup-A</Subject>Subjects> example inside <Rule>: <Subjects> <Subject type="X500Name">/O=Nordu-Grid/OU=UIO/CN=test</Subject> <Subject type="string">/vo.knowarc/usergroupA</Subject> <Subject> <SubFraction type="string">/O=Grid/OU=KnowARC/CN=XYZ</SubFraction> <SubFraction type="string">urn:mace:shibboleth:examples</SubFraction> </Subject> <GroupIdRef location="/.subjectgroup.xml">subgrpexample1</GroupIdRef> </Subjects>

### 4.1.2.2 typedef std::list<[Match](#)> [ArcSec::AndList](#)

AndList - include items inside one <Subject> (or <Resource> <Action> <Condition>).

"Or" relationship meand the request should satisfy any of the items <Subjects>  
 <Subject type="X500DN">/O=Grid/OU=KnowARC/CN=ABC</Subject> <Subject  
 type="VOMSAttribute">/vo.knowarc/usergroupA</Subject> <Subject> <SubFraction  
 type="X500DN">/O=Grid/OU=KnowARC/CN=XYZ</SubFraction> <SubFraction  
 type="ShibName">urn:mace:shibboleth:examples</SubFraction> </Subject> <GroupIdRef  
 location="/.subjectgroup.xml">subgrpexample1</GroupIdRef> </Subjects>

### 4.1.2.3 typedef std::list<[AndList](#)> [ArcSec::OrList](#)

OrList - include items inside one <Subjects> (or <Resources> <Actions> <Conditions>).



## Chapter 5

# Hosting Environment (Daemon) Chain Components Data Structure Documentation

### 5.1 ArcSec::AllowPDP Class Reference

This PDP always return true (allow).

```
#include <AllowPDP.h>
```

#### 5.1.1 Detailed Description

This PDP always return true (allow).

The documentation for this class was generated from the following file:

- AllowPDP.h

## 5.2 ArcSec::ArcAlgFactory Class Reference

Algorithm factory class for Arc.

```
#include <ArcAlgFactory.h>
```

### Public Member Functions

- virtual CombiningAlg \* [createAlg](#) (const std::string &type)

#### 5.2.1 Detailed Description

Algorithm factory class for Arc.

#### 5.2.2 Member Function Documentation

**5.2.2.1** virtual CombiningAlg\* ArcSec::ArcAlgFactory::createAlg (const std::string & *type*)  
[virtual]

return a Alg object according to the "CombiningAlg" attribute in the <Policy> node; The [ArcAlgFactory](#) itself will release the Alg objects

The documentation for this class was generated from the following file:

- ArcAlgFactory.h

## 5.3 ArcSec::ArcAttributeFactory Class Reference

Attribute factory class for Arc specified attributes.

```
#include <ArcAttributeFactory.h>
```

### Public Member Functions

- virtual AttributeValue \* [createValue](#) (const Arc::XMLNode &node, const std::string &type)

#### 5.3.1 Detailed Description

Attribute factory class for Arc specified attributes.

#### 5.3.2 Member Function Documentation

**5.3.2.1** `virtual AttributeValue* ArcSec::ArcAttributeFactory::createValue (const Arc::XMLNode & node, const std::string & type) [virtual]`

creat a AttributeValue according to the value in the XML node and the type; It should be the caller to release the AttributeValue Object

The documentation for this class was generated from the following file:

- ArcAttributeFactory.h

## 5.4 ArcSec::ArcAttributeProxy< TheAttribute > Class Template Reference

Arc specific AttributeProxy class.

```
#include <ArcAttributeProxy.h>
```

### Public Member Functions

- virtual AttributeValue \* [getAttribute](#) (const Arc::XMLNode &node)

#### 5.4.1 Detailed Description

```
template<class TheAttribute> class ArcSec::ArcAttributeProxy< TheAttribute >
```

Arc specific AttributeProxy class.

#### 5.4.2 Member Function Documentation

**5.4.2.1** `template<class TheAttribute> AttributeValue * ArcSec::ArcAttributeProxy< TheAttribute >::getAttribute (const Arc::XMLNode & node) [virtual]`

Implementation of getAttribute method.

The documentation for this class was generated from the following file:

- ArcAttributeProxy.h

## 5.5 ArcSec::ArcAuthZ Class Reference

Tests message against list of PDPs.

```
#include <ArcAuthZ.h>
```

### Public Member Functions

- virtual bool [Handle](#) (Arc::Message \*msg) const

### Protected Member Functions

- bool [MakePDPs](#) (Arc::XMLNode cfg)

### Data Structures

- class **PDPDesc**

#### 5.5.1 Detailed Description

Tests message against list of PDPs.

This class implements SecHandler interface. It's [Handle\(\)](#) method runs provided Message instance against all PDPs specified in configuration. If any of PDPs returns positive result [Handle\(\)](#) return true, otherwise false. This class is the main entry for configuring authorization, and could include different PDP configured inside.

#### 5.5.2 Member Function Documentation

##### 5.5.2.1 virtual bool ArcSec::ArcAuthZ::Handle (Arc::Message \* msg) const [virtual]

Get authorization decision

##### 5.5.2.2 bool ArcSec::ArcAuthZ::MakePDPs (Arc::XMLNode cfg) [protected]

Create PDP according to conf info

The documentation for this class was generated from the following file:

- ArcAuthZ.h

## 5.6 ArcSec::ArcEvaluationCtx Class Reference

EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc.

```
#include <ArcEvaluationCtx.h>
```

### Public Member Functions

- [ArcEvaluationCtx](#) (Request \*request)
- virtual void [split](#) ()

#### 5.6.1 Detailed Description

EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc.

#### 5.6.2 Constructor & Destructor Documentation

##### 5.6.2.1 ArcSec::ArcEvaluationCtx::ArcEvaluationCtx (Request \* *request*)

Construct a new EvaluationCtx based on the given request

#### 5.6.3 Member Function Documentation

##### 5.6.3.1 virtual void ArcSec::ArcEvaluationCtx::split () [virtual]

Convert/split one RequestItem ( one tuple <SubList, ResList, ActList, CtxList>) into a few <Subject, Resource, Action, Context> tuples. The purpose is for evaluation. The evaluator will evaluate each Request-Tuple one by one, not the RequestItem because it includes some independent <Subject, Resource, Action, Context>s and the evaluator should deal with them independently.

The documentation for this class was generated from the following file:

- ArcEvaluationCtx.h

## 5.7 ArcSec::ArcEvaluator Class Reference

Execute the policy evaluation, based on the request and policy.

```
#include <ArcEvaluator.h>
```

### Public Member Functions

- virtual Response \* [evaluate](#) (Request \*request)

#### 5.7.1 Detailed Description

Execute the policy evaluation, based on the request and policy.

#### 5.7.2 Member Function Documentation

##### 5.7.2.1 virtual Response\* ArcSec::ArcEvaluator::evaluate (Request \* *request*) [virtual]

Evaluate the request based on the policy information inside PolicyStore

The documentation for this class was generated from the following file:

- ArcEvaluator.h

## 5.8 ArcSec::ArcFnFactory Class Reference

Function factory class for Arc specified attributes.

```
#include <ArcFnFactory.h>
```

### Public Member Functions

- virtual Function \* [createFn](#) (const std::string &type)

#### 5.8.1 Detailed Description

Function factory class for Arc specified attributes.

#### 5.8.2 Member Function Documentation

##### 5.8.2.1 virtual Function\* ArcSec::ArcFnFactory::createFn (const std::string & type) [virtual]

return a Function object according to the "Function" attribute in the XML node; The [ArcFnFactory](#) itself will release the Function objects

The documentation for this class was generated from the following file:

- ArcFnFactory.h



## 5.9 ArcSec::ArcPDP Class Reference

[ArcPDP](#) - PDP which can handle the Arc specific request and policy schema.

```
#include <ArcPDP.h>
```

### 5.9.1 Detailed Description

[ArcPDP](#) - PDP which can handle the Arc specific request and policy schema.

The documentation for this class was generated from the following file:

- ArcPDP.h

## 5.10 ArcSec::ArcPolicy Class Reference

[ArcPolicy](#) class to parse and operate Arc specific <Policy> node.

```
#include <ArcPolicy.h>
```

### Public Member Functions

- [ArcPolicy](#) (Arc::PluginArgument \*parg)
- [ArcPolicy](#) (const Arc::XMLNode node, Arc::PluginArgument \*parg)
- [ArcPolicy](#) (const Arc::XMLNode node, EvaluatorContext \*ctx, Arc::PluginArgument \*parg)
- virtual void [make\\_policy](#) ()

### 5.10.1 Detailed Description

[ArcPolicy](#) class to parse and operate Arc specific <Policy> node.

### 5.10.2 Constructor & Destructor Documentation

#### 5.10.2.1 ArcSec::ArcPolicy::ArcPolicy (Arc::PluginArgument \* *parg*)

Constructor

#### 5.10.2.2 ArcSec::ArcPolicy::ArcPolicy (const Arc::XMLNode *node*, Arc::PluginArgument \* *parg*)

Constructor

#### 5.10.2.3 ArcSec::ArcPolicy::ArcPolicy (const Arc::XMLNode *node*, EvaluatorContext \* *ctx*, Arc::PluginArgument \* *parg*)

Constructor

### 5.10.3 Member Function Documentation

#### 5.10.3.1 virtual void ArcSec::ArcPolicy::make\_policy () [virtual]

Parse XMLNode, and construct the low-level Rule object

The documentation for this class was generated from the following file:

- ArcPolicy.h

## 5.11 ArcSec::ArcRequestItem Class Reference

Container, <Subjects, Actions, Objects, Contexts> tuple.

```
#include <ArcRequestItem.h>
```

### 5.11.1 Detailed Description

Container, <Subjects, Actions, Objects, Contexts> tuple.

Specified [ArcRequestItem](#) which can parse Arc request formate

The documentation for this class was generated from the following file:

- ArcRequestItem.h

## 5.12 ArcSec::ArcRequestTuple Class Reference

RequestTuple, container which includes the.

```
#include <ArcEvaluationCtx.h>
```

### 5.12.1 Detailed Description

RequestTuple, container which includes the.

The documentation for this class was generated from the following file:

- ArcEvaluationCtx.h

## 5.13 ArcSec::ArcRule Class Reference

[ArcRule](#) class to parse Arc specific <Rule> node.

```
#include <ArcRule.h>
```

### 5.13.1 Detailed Description

[ArcRule](#) class to parse Arc specific <Rule> node.

The documentation for this class was generated from the following file:

- ArcRule.h

## 5.14 Arc::DataPointARC Class Reference

```
#include <DataPointARC.h>
```

### 5.14.1 Detailed Description

Provides an interface to the Chelonia storage system developed by ARC.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointARC.h

## 5.15 Arc::DataPointFile Class Reference

```
#include <DataPointFile.h>
```

### 5.15.1 Detailed Description

This class allows access to the regular local filesystem through the same interface as is used for remote storage on the grid.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointFile.h

## 5.16 Arc::DataPointGFAL Class Reference

```
#include <DataPointGFAL.h>
```

### 5.16.1 Detailed Description

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointGFAL.h



## 5.17 Arc::DataPointGridFTP Class Reference

```
#include <DataPointGridFTP.h>
```

### Data Structures

- class **CBArg**

#### 5.17.1 Detailed Description

GridFTP is essentially the FTP protocol with GSI security. This class uses libraries from the Globus Toolkit. It can also be used for regular FTP.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointGridFTP.h

## 5.18 Arc::DataPointHTTP Class Reference

```
#include <DataPointHTTP.h>
```

### 5.18.1 Detailed Description

This class allows access through HTTP to remote resources. HTTP over SSL (HTTPS) and HTTP over GSI (HTTPG) are also supported.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointHTTP.h

## 5.19 Arc::DataPointLDAP Class Reference

```
#include <DataPointLDAP.h>
```

### 5.19.1 Detailed Description

LDAP is used in grids mainly to store information about grid services or resources rather than to store data itself. This class allows access to LDAP data through the same interface as other grid resources.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointLDAP.h

## 5.20 Arc::DataPointLFC Class Reference

```
#include <DataPointLFC.h>
```

### 5.20.1 Detailed Description

The LCG File Catalog (LFC) is a replica catalog developed by CERN. It consists of a hierarchical namespace of grid files and each filename can be associated with one or more physical locations.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointLFC.h

## 5.21 Arc::DataPointRLS Class Reference

```
#include <DataPointRLS.h>
```

### 5.21.1 Detailed Description

The Replica Location Service (RLS) is a replica catalog developed by Globus. It maps filenames in a flat namespace to one or more physical locations, and can also store meta-information on each file. This class uses the Globus Toolkit libraries for accessing RLS.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointRLS.h

## 5.22 Arc::DataPointSRM Class Reference

```
#include <DataPointSRM.h>
```

### 5.22.1 Detailed Description

The Storage Resource Manager (SRM) protocol allows access to data distributed across physical storage through a unified namespace and management interface. PrepareReading() or PrepareWriting() must be used before reading or writing a physical file.

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointSRM.h

## 5.23 Arc::DataPointXrootd Class Reference

```
#include <DataPointXrootd.h>
```

### 5.23.1 Detailed Description

xrootd is a protocol for data access across large scale storage clusters. More information can be found at <http://xrootd.slac.stanford.edu/>

This class is a loadable module and cannot be used directly. The DataHandle class loads modules at runtime and should be used instead of this.

The documentation for this class was generated from the following file:

- DataPointXrootd.h

## 5.24 ArcSec::DelegationPDP Class Reference

```
#include <DelegationPDP.h>
```

### 5.24.1 Detailed Description

DeleagtionPDP - PDP which can handle the Arc specific request and policy provided as identity delegation policy.

The documentation for this class was generated from the following file:

- DelegationPDP.h



## 5.25 ArcSec::DenyPDP Class Reference

This PDP always returns false (deny).

```
#include <DenyPDP.h>
```

### 5.25.1 Detailed Description

This PDP always returns false (deny).

The documentation for this class was generated from the following file:

- DenyPDP.h

## 5.26 Arc::LDAPQuery Class Reference

```
#include <LDAPQuery.h>
```

### Public Member Functions

- [LDAPQuery](#) (const std::string &ldaphost, int ldapport, int timeout, bool anonymous=true, const std::string &usersn="")
- [~LDAPQuery](#) ()
- bool [Query](#) (const std::string &base, const std::string &filter="(objectclass=\*)", const std::list< std::string > &attributes=std::list< std::string >(), URL::Scope scope=URL::subtree)
- bool [Result](#) (ldap\_callback callback, void \*ref)

### 5.26.1 Detailed Description

[LDAPQuery](#) class; querying of LDAP servers.

### 5.26.2 Constructor & Destructor Documentation

#### 5.26.2.1 Arc::LDAPQuery::LDAPQuery (const std::string & ldaphost, int ldapport, int timeout, bool anonymous = true, const std::string & usersn = "")

Constructs a new [LDAPQuery](#) object and sets connection options. The connection is first established when calling [Query](#).

#### 5.26.2.2 Arc::LDAPQuery::~~LDAPQuery ()

Destructor. Will disconnect from the ldapserver if still connected.

### 5.26.3 Member Function Documentation

#### 5.26.3.1 bool Arc::LDAPQuery::Query (const std::string & base, const std::string & filter = "(objectclass=\*)", const std::list< std::string > & attributes = std::list< std::string >(), URL::Scope scope = URL::subtree)

Queries the ldap server.

#### 5.26.3.2 bool Arc::LDAPQuery::Result (ldap\_callback callback, void \* ref)

Retrieves the result of the query from the ldap-server.

The documentation for this class was generated from the following file:

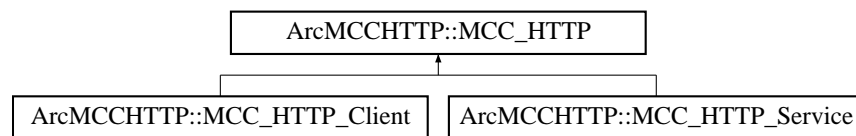
- LDAPQuery.h

## 5.27 ArcMCCHTTP::MCC\_HTTP Class Reference

A base class for HTTP client and service MCCs.

```
#include <MCCHTTP.h>
```

Inheritance diagram for ArcMCCHTTP::MCC\_HTTP::



### 5.27.1 Detailed Description

A base class for HTTP client and service MCCs.

This is a base class for HTTP client and service MCCs. It provides some common functionality for them, i.e. so far only a logger.

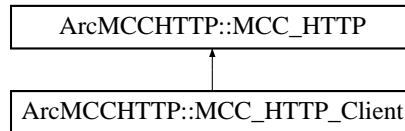
The documentation for this class was generated from the following file:

- MCCHTTP.h

## 5.28 ArcMCCHTTP::MCC\_HTTP\_Client Class Reference

```
#include <MCCHTTP.h>
```

Inheritance diagram for ArcMCCHTTP::MCC\_HTTP\_Client::



### 5.28.1 Detailed Description

This class is a client part of HTTP MCC. It accepts PayloadRawInterface payload and uses it as body to generate HTTP request. Request is passed to next MCC as PayloadRawInterface type of payload. Returned PayloadStreamInterface payload is parsed into HTTP response and it's body is passed back to calling MCC as PayloadRawInterface. Attributes of request/input message of type HTTP:name are translated into HTTP header with corresponding 'name's. Special attributes HTTP:METHOD and HTTP:ENDPOINT specify method and URL in HTTP request. If not present meathod and URL are taken from configuration. In output/response message following attributes are present: HTTP:CODE - response code of HTTP HTTP:REASON - reason string of HTTP response HTTP:name - all 'name' attributes of HTTP header.

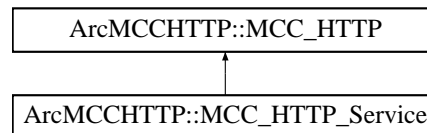
The documentation for this class was generated from the following file:

- MCCHTTP.h

## 5.29 ArcMCCHTTP::MCC\_HTTP\_Service Class Reference

```
#include <MCCHTTP.h>
```

Inheritance diagram for ArcMCCHTTP::MCC\_HTTP\_Service::



### 5.29.1 Detailed Description

This class implements MCC to processes HTTP request. On input payload with PayloadStreamInterface is expected. HTTP message is read from stream and its body is converted into PayloadRaw and passed to next MCC. Returned payload of PayloadRawInterface type is treated as body part of returning [Payload-HTTP](#). Generated HTTP response is sent through stream passed in input payload. During processing of request/input message following attributes are generated: HTTP:METHOD - HTTP method e.g. GET, PUT, POST, etc. HTTP:ENDPOINT - URL taken from HTTP request ENDPOINT - global attribute equal to HTTP:ENDPOINT HTTP:RANGESTART - start of requested byte range HTTP:RANGEEND - end of requested byte range (inclusive) HTTP:name - all 'name' attributes of HTTP header. Attributes of response message of HTTP:name type are translated into HTTP header with corresponding 'name's.

The documentation for this class was generated from the following file:

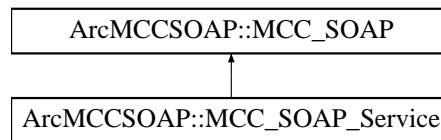
- MCCHTTP.h

## 5.30 ArcMCCSOAP::MCC\_SOAP Class Reference

A base class for SOAP client and service MCCs.

```
#include <MCCSOAP.h>
```

Inheritance diagram for ArcMCCSOAP::MCC\_SOAP::



### 5.30.1 Detailed Description

A base class for SOAP client and service MCCs.

This is a base class for SOAP client and service MCCs. It provides some common functionality for them, i.e. so far only a logger.

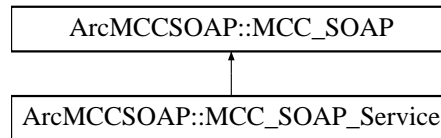
The documentation for this class was generated from the following file:

- MCCSOAP.h

## 5.31 ArcMCCSOAP::MCC\_SOAP\_Service Class Reference

```
#include <MCCSOAP.h>
```

Inheritance diagram for ArcMCCSOAP::MCC\_SOAP\_Service::



### 5.31.1 Detailed Description

This MCC parses SOAP message from input payload. On input payload with PayloadRawInterface is expected. It's converted into PayloadSOAP and passed next MCC. Returned PayloadSOAP is converted into PayloadRaw and returned to calling MCC.

The documentation for this class was generated from the following file:

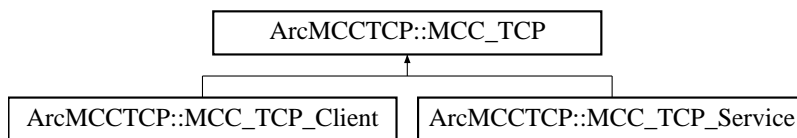
- MCCSOAP.h

## 5.32 ArcMCCTCP::MCC\_TCP Class Reference

A base class for TCP client and service MCCs.

```
#include <MCCTCP.h>
```

Inheritance diagram for ArcMCCTCP::MCC\_TCP::



### 5.32.1 Detailed Description

A base class for TCP client and service MCCs.

This is a base class for TCP client and service MCCs. It provides some common functionality for them, i.e. so far only a logger.

The documentation for this class was generated from the following file:

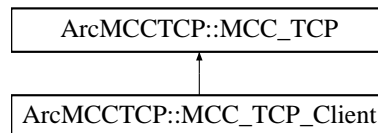
- MCCTCP.h



## 5.33 ArcMCCTCP::MCC\_TCP\_Client Class Reference

```
#include <MCCTCP.h>
```

Inheritance diagram for ArcMCCTCP::MCC\_TCP\_Client::



### 5.33.1 Detailed Description

This class is MCC implementing TCP client. Upon creation it connects to specified TCP post at specified host. process() method accepts PayloadRawInterface type of payload. Content of payload is sent over TCP socket. It returns PayloadStreamInterface payload for previous MCC to read response.

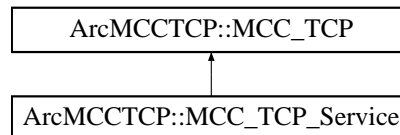
The documentation for this class was generated from the following file:

- MCCTCP.h

## 5.34 ArcMCCTCP::MCC\_TCP\_Service Class Reference

```
#include <MCCTCP.h>
```

Inheritance diagram for ArcMCCTCP::MCC\_TCP\_Service::



### Public Member Functions

- [MCC\\_TCP\\_Service](#) (Config \*cfg, PluginArgument \*parg)

### Data Structures

- class `mcc_tcp_exec_t`
- class `mcc_tcp_handle_t`

#### 5.34.1 Detailed Description

This class is MCC implementing TCP server. Upon creation this object binds to specified TCP ports and listens for incoming TCP connections on dedicated thread. Each connection is accepted and dedicated thread is created. Then that thread is used to call process() method of next MCC in chain. That method is passed payload implementing PayloadStreamInterface. On response payload with PayloadRawInterface is expected. Alternatively called MCC may use provided PayloadStreamInterface to send it's response back directly. During processing of request this MCC generates following attributes: TCP:HOST - IP address of interface to which local TCP socket is bound TCP:PORT - port number to which local TCP socket is bound TCP:REMOTEHOST - IP address from which connection is accepted TCP:REMOTEPORT - TCP port from which connection is accepted TCP:ENDPOINT - URL-like representation of remote connection - `://HOST:PORT ENDPOINT` - global attribute equal to TCP:ENDPOINT

#### 5.34.2 Constructor & Destructor Documentation

##### 5.34.2.1 ArcMCCTCP::MCC\_TCP\_Service::MCC\_TCP\_Service (Config \* *cfg*, PluginArgument \* *parg*)

executing function for connection thread

The documentation for this class was generated from the following file:

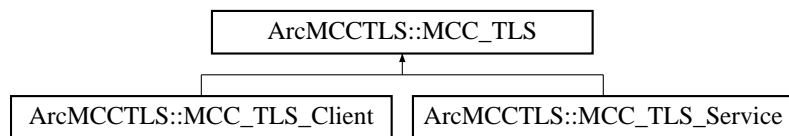
- MCCTCP.h

## 5.35 ArcMCCTLS::MCC\_TLS Class Reference

A base class for TLS client and service MCCs.

```
#include <MCCTLS.h>
```

Inheritance diagram for ArcMCCTLS::MCC\_TLS::



### 5.35.1 Detailed Description

A base class for TLS client and service MCCs.

This is a base class for TLS client and service MCCs. It provides some common functionality for them.

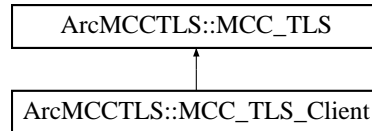
The documentation for this class was generated from the following file:

- MCCTLS.h

## 5.36 ArcMCCTLS::MCC\_TLS\_Client Class Reference

```
#include <MCCTLS.h>
```

Inheritance diagram for ArcMCCTLS::MCC\_TLS\_Client::



### 5.36.1 Detailed Description

This class is MCC implementing TLS client.

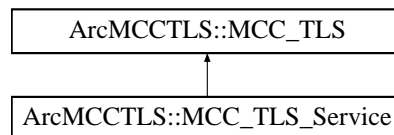
The documentation for this class was generated from the following file:

- MCCTLS.h

## 5.37 ArcMCCTLS::MCC\_TLS\_Service Class Reference

```
#include <MCCTLS.h>
```

Inheritance diagram for ArcMCCTLS::MCC\_TLS\_Service::



### 5.37.1 Detailed Description

This MCC implements TLS server side functionality. Upon creation this object creates SSL\_CTX object and configures SSL\_CTX object with some environment information about credential. Because we cannot know the "socket" when the creation of MCC\_TLS\_Service/MCC\_TLS\_Client object (not like MCC\_TCP\_Client, which can creat socket in the constructor method by using information in configuration file), we can only creat "ssl" object which is binded to specified "socket", when MCC\_HTTP\_Client calls the process() method of [MCC\\_TLS\\_Client](#) object, or MCC\_TCP\_Service calls the process() method of [MCC\\_TLS\\_Service](#) object. The "ssl" object is embeded in a payload called PayloadTLSSocket.

The process() method of [MCC\\_TLS\\_Service](#) is passed payload implementing PayloadStreamInterface and the method returns empty PayloadRaw payload in "outmsg". The ssl object is created and bound to Stream payload when constructing the PayloadTLSSocket in the process() method.

During processing of message this MCC generates attribute TLS:PEERDN which contains Distinguished Name of remoote peer.

The documentation for this class was generated from the following file:

- MCCTLS.h

## 5.38 ArcMCCHTTP::PayloadHTTP Class Reference

```
#include <PayloadHTTP.h>
```

### Public Member Functions

- [PayloadHTTP](#) (PayloadStreamInterface &stream, bool own=false)
- [PayloadHTTP](#) (const std::string &method, const std::string &url, PayloadStreamInterface &stream)
- [PayloadHTTP](#) (const std::string &method, const std::string &url)
- [PayloadHTTP](#) (int code, const std::string &reason, PayloadStreamInterface &stream, bool head\_response=false)
- [PayloadHTTP](#) (int code, const std::string &reason, bool head\_response=false)
- virtual const std::string & [Attribute](#) (const std::string &name)
- virtual const std::multimap< std::string, std::string > & [Attributes](#) (void)
- virtual void [Attribute](#) (const std::string &name, const std::string &value)
- virtual bool [Flush](#) (void)
- virtual void [Body](#) (PayloadRawInterface &body, bool ownership=true)

### Protected Member Functions

- bool [readline](#) (std::string &line)
- bool [read](#) (char \*buf, int64\_t &size)
- bool [read\\_header](#) (void)
- bool [get\\_body](#) (void)

### Protected Attributes

- PayloadStreamInterface \* [stream\\_](#)
- bool [stream\\_own\\_](#)
- PayloadRawInterface \* [rbody\\_](#)
- PayloadStreamInterface \* [sbody\\_](#)
- bool [body\\_own\\_](#)
- std::string [uri\\_](#)
- int [version\\_major\\_](#)
- int [version\\_minor\\_](#)
- std::string [method\\_](#)
- int [code\\_](#)
- std::string [reason\\_](#)
- int64\_t [length\\_](#)
- int64\_t [end\\_](#)
- chunked\_t [chunked\\_](#)
- int64\_t [chunk\\_size\\_](#)
- std::multimap< std::string, std::string > [attributes\\_](#)

#### 5.38.1 Detailed Description

This class implements parsing and generation of HTTP messages. It implements only subset of HTTP/1.1 and also provides an PayloadRawInterface for including as payload into Message passed through MCC chains.

## 5.38.2 Constructor & Destructor Documentation

### 5.38.2.1 ArcMCCHTTP::PayloadHTTP::PayloadHTTP (PayloadStreamInterface & *stream*, bool *own* = false)

Constructor - creates object by parsing HTTP request or response from stream. Supplied stream is associated with object for later use. If *own* is set to true then stream will be deleted in destructor. Because stream can be used by this object during whole lifetime it is important not to destroy stream till this object is deleted.

### 5.38.2.2 ArcMCCHTTP::PayloadHTTP::PayloadHTTP (const std::string & *method*, const std::string & *url*, PayloadStreamInterface & *stream*)

Constructor - creates HTTP request to be sent through stream. HTTP message is not sent yet.

### 5.38.2.3 ArcMCCHTTP::PayloadHTTP::PayloadHTTP (const std::string & *method*, const std::string & *url*)

Constructor - creates HTTP request to be rendered through Raw interface.

### 5.38.2.4 ArcMCCHTTP::PayloadHTTP::PayloadHTTP (int *code*, const std::string & *reason*, PayloadStreamInterface & *stream*, bool *head\_response* = false)

Constructor - creates HTTP response to be sent through stream. HTTP message is not sent yet.

### 5.38.2.5 ArcMCCHTTP::PayloadHTTP::PayloadHTTP (int *code*, const std::string & *reason*, bool *head\_response* = false)

Constructor - creates HTTP response to be rendered through Raw interface.

## 5.38.3 Member Function Documentation

### 5.38.3.1 virtual void ArcMCCHTTP::PayloadHTTP::Attribute (const std::string & *name*, const std::string & *value*) [virtual]

Adds HTTP header attribute 'name' = 'value'

### 5.38.3.2 virtual const std::string& ArcMCCHTTP::PayloadHTTP::Attribute (const std::string & *name*) [virtual]

Returns HTTP header attribute with specified name. Empty string if no such attribute.

### 5.38.3.3 virtual const std::multimap<std::string, std::string>& ArcMCCHTTP::PayloadHTTP::Attributes (void) [virtual]

Returns all HTTP header attributes.

#### 5.38.3.4 **virtual void ArcMCCHTTP::PayloadHTTP::Body (PayloadRawInterface & *body*, bool *ownership* = true) [virtual]**

Assign HTTP body. Assigned object is not copied. Instead it is remembered and made available through Raw interface. If 'ownership' is true then passed object is treated as being owned by this instance and destroyed in destructor.

#### 5.38.3.5 **virtual bool ArcMCCHTTP::PayloadHTTP::Flush (void) [virtual]**

Send created object through associated stream. If there is no stream associated then HTTP specific data is inserted into Raw buffers of this object. In last case this operation should not be repeated till content of buffer is completely rewritten.

#### 5.38.3.6 **bool ArcMCCHTTP::PayloadHTTP::get\_body (void) [protected]**

Read Body of HTTP message and attach it to inherited PayloadRaw object

#### 5.38.3.7 **bool ArcMCCHTTP::PayloadHTTP::read (char \* *buf*, int64\_t & *size*) [protected]**

Read up to 'size' bytes from stream\_

#### 5.38.3.8 **bool ArcMCCHTTP::PayloadHTTP::read\_header (void) [protected]**

Read HTTP header and fill internal variables

#### 5.38.3.9 **bool ArcMCCHTTP::PayloadHTTP::readline (std::string & *line*) [protected]**

Read from stream till

### 5.38.4 **Field Documentation**

#### 5.38.4.1 **std::multimap<std::string, std::string> ArcMCCHTTP::PayloadHTTP::attributes\_ [protected]**

true if conection should not be closed after response

#### 5.38.4.2 **bool ArcMCCHTTP::PayloadHTTP::body\_own\_ [protected]**

associated HTTP Body stream if any (to avoid copying to own buffer)

#### 5.38.4.3 **int64\_t ArcMCCHTTP::PayloadHTTP::chunk\_size\_ [protected]**

chunked encoding parsing state

#### 5.38.4.4 **chunked\_t ArcMCCHTTP::PayloadHTTP::chunked\_ [protected]**

Logical end of content computed from Content-Range



**5.38.4.5** `int ArcMCCHTTP::PayloadHTTP::code_` [protected]

HTTP method being used or requested

**5.38.4.6** `int64_t ArcMCCHTTP::PayloadHTTP::end_` [protected]

Content-length of HTTP message

**5.38.4.7** `int64_t ArcMCCHTTP::PayloadHTTP::length_` [protected]

HTTP reason being sent or supplied

**5.38.4.8** `std::string ArcMCCHTTP::PayloadHTTP::method_` [protected]

minor number of HTTP version - must be 0 or 1

**5.38.4.9** `PayloadRawInterface* ArcMCCHTTP::PayloadHTTP::rbody_` [protected]

if true stream\_ is owned by this

**5.38.4.10** `std::string ArcMCCHTTP::PayloadHTTP::reason_` [protected]

HTTP code being sent or supplied

**5.38.4.11** `PayloadStreamInterface* ArcMCCHTTP::PayloadHTTP::sbody_` [protected]

associated HTTP Body buffer if any (to avoid copying to own buffer)

**5.38.4.12** `PayloadStreamInterface* ArcMCCHTTP::PayloadHTTP::stream_` [protected]

true if whole content of HTTP body was fetched and stored in buffers. Otherwise only header was fetched and part of body in tbuf\_ and rest is to be read through stream\_.

**5.38.4.13** `bool ArcMCCHTTP::PayloadHTTP::stream_own_` [protected]

stream used to communicate to outside

**5.38.4.14** `std::string ArcMCCHTTP::PayloadHTTP::uri_` [protected]

if true body\_ is owned by this

**5.38.4.15** `int ArcMCCHTTP::PayloadHTTP::version_major_` [protected]

URI being contacted

**5.38.4.16**   `int` [ArcMCCHTTP::PayloadHTTP::version\\_minor\\_](#)   [protected]

major number of HTTP version - must be 1

The documentation for this class was generated from the following file:

- PayloadHTTP.h

## 5.39 ArcMCCTCP::PayloadTCPSocket Class Reference

```
#include <PayloadTCPSocket.h>
```

### Public Member Functions

- [PayloadTCPSocket](#) (const char \*hostname, int port, int timeout, Logger &logger)
- [PayloadTCPSocket](#) (const std::string &endpoint, int timeout, Logger &logger)
- [PayloadTCPSocket](#) (int s, int timeout, Logger &logger)
- [PayloadTCPSocket](#) ([PayloadTCPSocket](#) &s)
- [PayloadTCPSocket](#) ([PayloadTCPSocket](#) &s, Logger &logger)

### 5.39.1 Detailed Description

This class extends PayloadStream with TCP socket specific features

### 5.39.2 Constructor & Destructor Documentation

#### 5.39.2.1 ArcMCCTCP::PayloadTCPSocket::PayloadTCPSocket (const char \* *hostname*, int *port*, int *timeout*, Logger & *logger*)

Constructor - connects to TCP server at specified hostname:port

#### 5.39.2.2 ArcMCCTCP::PayloadTCPSocket::PayloadTCPSocket (const std::string & *endpoint*, int *timeout*, Logger & *logger*)

Constructor - connects to TCP server at specified endpoint - hostname:port

#### 5.39.2.3 ArcMCCTCP::PayloadTCPSocket::PayloadTCPSocket (int *s*, int *timeout*, Logger & *logger*) [inline]

Constructor - creates object of already connected socket. Socket is NOT closed in destructor.

#### 5.39.2.4 ArcMCCTCP::PayloadTCPSocket::PayloadTCPSocket ([PayloadTCPSocket](#) & *s*) [inline]

Copy constructor - inherits socket of copied object. Socket is NOT closed in destructor.

#### 5.39.2.5 ArcMCCTCP::PayloadTCPSocket::PayloadTCPSocket ([PayloadTCPSocket](#) & *s*, Logger & *logger*) [inline]

Copy constructor - inherits handle of copied object. Handle is NOT closed in destructor.

The documentation for this class was generated from the following file:

- PayloadTCPSocket.h

## 5.40 ArcMCCTLS::PayloadTLSStream Class Reference

```
#include <PayloadTLSStream.h>
```

### Public Member Functions

- [PayloadTLSStream](#) (Logger &logger, SSL \*ssl=NULL)
- virtual [~PayloadTLSStream](#) (void)
- X509 \* [GetPeerCert](#) (void)
- [STACK\\_OF](#) (X509)\*GetPeerChain(void)
- X509 \* [GetCert](#) (void)

### Protected Attributes

- SSL \* [ssl\\_](#)

#### 5.40.1 Detailed Description

Implementation of PayloadStreamInterface for SSL handle.

#### 5.40.2 Constructor & Destructor Documentation

##### 5.40.2.1 ArcMCCTLS::PayloadTLSStream::PayloadTLSStream (Logger & *logger*, SSL \* *ssl* = NULL)

Constructor. Attaches to already open handle. Handle is not managed by this class and must be closed by external code.

##### 5.40.2.2 virtual ArcMCCTLS::PayloadTLSStream::~~PayloadTLSStream (void) [virtual]

Destructor.

#### 5.40.3 Member Function Documentation

##### 5.40.3.1 X509\* ArcMCCTLS::PayloadTLSStream::GetCert (void)

Get local certificate from associated ssl. Obtained X509 object is owned by this instance and becomes invalid after destruction.

##### 5.40.3.2 X509\* ArcMCCTLS::PayloadTLSStream::GetPeerCert (void)

Get peer certificate from the established ssl. Obtained X509 object is owned by this instance and becomes invalid after destruction. Still obtained has to be freed at end of usage.

### 5.40.3.3 ArcMCCTLS::PayloadTLSStream::STACK\_OF (X509)

Get chain of peer certificates from the established ssl. Obtained X509 object is owned by this instance and becomes invalid after destruction.

## 5.40.4 Field Documentation

### 5.40.4.1 SSL\* [ArcMCCTLS::PayloadTLSStream::ssl\\_](#) [protected]

Timeout for read/write operations

The documentation for this class was generated from the following file:

- PayloadTLSStream.h

## 5.41 ArcSec::PDPSERVICEInvoker Class Reference

[PDPSERVICEInvoker](#) - client which will invoke pdpservice.

```
#include <PDPSERVICEInvoker.h>
```

### 5.41.1 Detailed Description

[PDPSERVICEInvoker](#) - client which will invoke pdpservice.

The documentation for this class was generated from the following file:

- PDPSERVICEInvoker.h

## 5.42 ArcSec::SAML2SSO\_AssertionConsumerSH Class Reference

Implement the functionality of the Service Provider in SAML2 SSO profile.

```
#include <SAML2SSO_AssertionConsumerSH.h>
```

### 5.42.1 Detailed Description

Implement the functionality of the Service Provider in SAML2 SSO profile.

The documentation for this class was generated from the following file:

- SAML2SSO\_AssertionConsumerSH.h

## 5.43 ArcSec::SAMLTokenSH Class Reference

Adds WS-Security SAML Token into SOAP Header.

```
#include <SAMLTokenSH.h>
```

### 5.43.1 Detailed Description

Adds WS-Security SAML Token into SOAP Header.

The documentation for this class was generated from the following file:

- SAMLTokenSH.h



## 5.44 ArcSec::SimpleListPDP Class Reference

Tests X509 subject against list of subjects in file.

```
#include <SimpleListPDP.h>
```

### 5.44.1 Detailed Description

Tests X509 subject against list of subjects in file.

This class implements PDP interface. It's isPermitted() method compares X509 subject of requestor obtained from TLS layer (TLS:PEERDN) to list of subjects (one per line) in external file. Location of file is defined by 'location' attribute of PDP configuration. Returns true if subject is present in list, otherwise false.

The documentation for this class was generated from the following file:

- SimpleListPDP.h

## 5.45 Arc::SRMClient Class Reference

```
#include <SRMClient.h>
```

### Public Member Functions

- virtual [~SRMClient](#) ()
- std::string [getVersion](#) () const
- virtual SRMReturnCode [ping](#) (std::string &[version](#), bool report\_error=true)=0
- virtual SRMReturnCode [getSpaceTokens](#) (std::list< std::string > &tokens, const std::string &description="")=0
- virtual SRMReturnCode [getRequestTokens](#) (std::list< std::string > &tokens, const std::string &description="")=0
- virtual SRMReturnCode [getTURLs](#) (SRMClientRequest &req, std::list< std::string > &urls)=0
- virtual SRMReturnCode [getTURLsStatus](#) (SRMClientRequest &req, std::list< std::string > &urls)=0
- virtual SRMReturnCode [requestBringOnline](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [requestBringOnlineStatus](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [putTURLs](#) (SRMClientRequest &req, std::list< std::string > &urls)=0
- virtual SRMReturnCode [putTURLsStatus](#) (SRMClientRequest &req, std::list< std::string > &urls)=0
- virtual SRMReturnCode [releaseGet](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [releasePut](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [release](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [abort](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [info](#) (SRMClientRequest &req, std::map< std::string, std::list< struct [SRMFileMetaData](#) > > &metadata)=0
- virtual SRMReturnCode [info](#) (SRMClientRequest &req, std::list< struct [SRMFileMetaData](#) > &metadata)=0
- virtual SRMReturnCode [remove](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [copy](#) (SRMClientRequest &req, const std::string &source)=0
- virtual SRMReturnCode [mkDir](#) (SRMClientRequest &req)=0
- virtual SRMReturnCode [checkPermissions](#) (SRMClientRequest &req)=0

### Static Public Member Functions

- static [SRMClient](#) \* [getInstance](#) (const UserConfig &usercfg, const std::string &url, bool &timedout)

### Protected Member Functions

- [SRMClient](#) (const UserConfig &usercfg, const SRMURL &url)
- SRMReturnCode [process](#) (const std::string &action, PayloadSOAP \*request, PayloadSOAP \*\*response)

## Protected Attributes

- std::string [service\\_endpoint](#)
- MCCCConfig [cfg](#)
- ClientSOAP \* [client](#)
- NS [ns](#)
- SRMImplementation [implementation](#)
- time\_t [user\\_timeout](#)
- std::string [version](#)

## Static Protected Attributes

- static Logger [logger](#)

### 5.45.1 Detailed Description

A client interface to the SRM protocol. Instances of SRM clients are created by calling the [getInstance\(\)](#) factory method. One client instance can be used to make many requests to the same server (with the same protocol version), but not multiple servers.

### 5.45.2 Constructor & Destructor Documentation

#### 5.45.2.1 Arc::SRMClient::SRMClient (const UserConfig & *usercfg*, const SRMURL & *url*) [protected]

Protected constructor.

#### 5.45.2.2 virtual Arc::SRMClient::~~SRMClient () [virtual]

Destructor

### 5.45.3 Member Function Documentation

#### 5.45.3.1 virtual SRMReturnCode Arc::SRMClient::abort ([SRMClientRequest](#) & *req*) [pure virtual]

Called in the case of failure during transfer or releasePut. Releases all TURLs involved in the transfer.

#### Parameters:

*req* The request object

#### Returns:

SRMReturnCode specifying outcome of operation

**5.45.3.2 virtual SRMReturnCode Arc::SRMClient::checkPermissions (SRMClientRequest & req)**  
[pure virtual]

Check permissions for the SURL in the request using the current credentials. req The request object

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.3 virtual SRMReturnCode Arc::SRMClient::copy (SRMClientRequest & req, const std::string & source)** [pure virtual]

Copy a file between two SRM storages.

**Parameters:**

*req* The request object

*source* The source SURL

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.4 static SRMClient\* Arc::SRMClient::getInstance (const UserConfig & usercfg, const std::string & url, bool & timedout)** [static]

Create an SRMClient instance. The instance will be a SRM v2.2 client unless another version is explicitly given in the url.

**Parameters:**

*usercfg* The user configuration.

*url* A SURL. A client connects to the service host derived from this SURL. All operations with a client instance must use SURLs with the same host as this one.

*timedout* Whether the connection timed out

**Returns:**

A pointer to an instance of SRMClient is returned, or NULL if it was not possible to create one.

**5.45.3.5 virtual SRMReturnCode Arc::SRMClient::getRequestTokens (std::list< std::string > & tokens, const std::string & description = "")** [pure virtual]

Returns a list of request tokens for the user calling the method which are still active requests, or the tokens corresponding to the token description, if given.

**Parameters:**

*tokens* The list filled by the service

*description* The user request description, which can be specified when the request is created

**Returns:**

SRMReturnCode specifying outcome of operation

### 5.45.3.6 virtual SRMReturnCode Arc::SRMClient::getSpaceTokens (std::list< std::string > & *tokens*, const std::string & *description* = "") [pure virtual]

Find the space tokens available to write to which correspond to the space token description, if given. The list of tokens is a list of numbers referring to the SRM internal definition of the spaces, not user-readable strings.

#### Parameters:

*tokens* The list filled by the service  
*description* The space token description

#### Returns:

SRMReturnCode specifying outcome of operation

### 5.45.3.7 virtual SRMReturnCode Arc::SRMClient::getTURLs (SRMClientRequest & *req*, std::list< std::string > & *urls*) [pure virtual]

If the user wishes to copy a file from somewhere, [getTURLs\(\)](#) is called to retrieve the transport URL(s) to copy the file from. It may be used synchronously or asynchronously, depending on the synchronous property of the request object. In the former case it will block until the TURLs are ready, in the latter case it will return after making the request and [getTURLsStatus\(\)](#) must be used to poll the request status if it was not completed.

#### Parameters:

*req* The request object  
*urls* A list of TURLs filled by the method

#### Returns:

SRMReturnCode specifying outcome of operation

### 5.45.3.8 virtual SRMReturnCode Arc::SRMClient::getTURLsStatus (SRMClientRequest & *req*, std::list< std::string > & *urls*) [pure virtual]

In the case where [getTURLs](#) was called asynchronously and the request was not completed, this method should be called to poll the status of the request. [getTURLs](#) must be called before this method and the request object must have ongoing request status.

#### Parameters:

*req* The request object. Status must be ongoing.  
*urls* A list of TURLs filled by the method if the request completed successfully

#### Returns:

SRMReturnCode specifying outcome of operation

### 5.45.3.9 std::string Arc::SRMClient::getVersion () const [inline]

Returns the version of the SRM protocol used by this instance

**5.45.3.10** `virtual SRMReturnCode Arc::SRMClient::info (SRMClientRequest & req, std::list< struct SRMFileMetaData > & metadata)` [pure virtual]

Returns information on a file stored in an SRM, such as file size, checksum and estimated access latency. If a directory is listed with recursion  $\geq 1$  then the list in metadata will contain the content of the directory.

**Parameters:**

*req* The request object

*metadata* A list of structs filled with file information.

**Returns:**

SRMReturnCode specifying outcome of operation

**See also:**

[SRMFileMetaData](#)

**5.45.3.11** `virtual SRMReturnCode Arc::SRMClient::info (SRMClientRequest & req, std::map< std::string, std::list< struct SRMFileMetaData > > & metadata)` [pure virtual]

Returns information on a file or files (v2.2 and higher) stored in SRM, such as file size, checksum and estimated access latency. If a directory or directories is listed with recursion  $\geq 1$  then the list mapped to each SURL in metadata will contain the content of the directory or directories.

**Parameters:**

*req* The request object

*metadata* A map mapping each SURL in the request to a list of structs filled with file information. If a SURL is missing from the map it means there was some problem accessing it.

**Returns:**

SRMReturnCode specifying outcome of operation

**See also:**

[SRMFileMetaData](#)

**5.45.3.12** `virtual SRMReturnCode Arc::SRMClient::mkDir (SRMClientRequest & req)` [pure virtual]

Make required directories for the SURL in the request

**Parameters:**

*req* The request object

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.13 virtual SRMReturnCode Arc::SRMClient::ping (std::string & *version*, bool *report\_error* = true) [pure virtual]**

Find out the version supported by the server this client is connected to. Since this method is used to determine which client version to instantiate, we may not want to report an error to the user, so setting *report\_error* to false suppresses the error message.

**Parameters:**

*version* The version returned by the server  
*report\_error* Whether an error should be reported

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.14 SRMReturnCode Arc::SRMClient::process (const std::string & *action*, PayloadSOAP \* *request*, PayloadSOAP \*\* *response*) [protected]**

Process SOAP request.

**5.45.3.15 virtual SRMReturnCode Arc::SRMClient::putURLs (SRMClientRequest & *req*, std::list< std::string > & *urls*) [pure virtual]**

If the user wishes to copy a file to somewhere, [putURLs\(\)](#) is called to retrieve the transport URL(s) to copy the file to. It may be used synchronously or asynchronously, depending on the synchronous property of the request object. In the former case it will block until the TURLs are ready, in the latter case it will return after making the request and [putURLsStatus\(\)](#) must be used to poll the request status if it was not completed.

**Parameters:**

*req* The request object  
*urls* A list of TURLs filled by the method

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.16 virtual SRMReturnCode Arc::SRMClient::putURLsStatus (SRMClientRequest & *req*, std::list< std::string > & *urls*) [pure virtual]**

In the case where [putURLs](#) was called asynchronously and the request was not completed, this method should be called to poll the status of the request. [putURLs](#) must be called before this method and the request object must have ongoing request status.

**Parameters:**

*req* The request object. Status must be ongoing.  
*urls* A list of TURLs filled by the method if the request completed successfully

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.17 virtual SRMReturnCode Arc::SRMClient::release (SRMClientRequest & req) [pure virtual]**

Used in SRM v1 only. Called to release files after successful transfer.

**Parameters:**

*req* The request object

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.18 virtual SRMReturnCode Arc::SRMClient::releaseGet (SRMClientRequest & req) [pure virtual]**

Should be called after a successful copy from SRM storage.

**Parameters:**

*req* The request object

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.19 virtual SRMReturnCode Arc::SRMClient::releasePut (SRMClientRequest & req) [pure virtual]**

Should be called after a successful copy to SRM storage.

**Parameters:**

*req* The request object

**Returns:**

SRMReturnCode specifying outcome of operation

**5.45.3.20 virtual SRMReturnCode Arc::SRMClient::remove (SRMClientRequest & req) [pure virtual]**

Delete a file physically from storage and the SRM namespace.

**Parameters:**

*req* The request object

**Returns:**

SRMReturnCode specifying outcome of operation



### 5.45.3.21 virtual SRMReturnCode Arc::SRMClient::requestBringOnline (SRMClientRequest & req) [pure virtual]

Submit a request to bring online files. If the synchronous property of the request object is false, this operation is asynchronous and the status of the request can be checked by calling [requestBringOnlineStatus\(\)](#) with the request token in req which is assigned by this method. If the request is synchronous, this operation blocks until the file(s) are online or the timeout specified in the [SRMClient](#) constructor has passed.

#### Parameters:

*req* The request object

#### Returns:

SRMReturnCode specifying outcome of operation

### 5.45.3.22 virtual SRMReturnCode Arc::SRMClient::requestBringOnlineStatus (SRMClientRequest & req) [pure virtual]

Query the status of a request to bring files online. The SURIs map of the request object is updated if the status of any files in the request has changed. [requestBringOnline\(\)](#) but be called before this method.

#### Parameters:

*req* The request object to query the status of

#### Returns:

SRMReturnCode specifying outcome of operation

## 5.45.4 Field Documentation

### 5.45.4.1 MCCCConfig Arc::SRMClient::cfg [protected]

SOAP configuraton object.

### 5.45.4.2 ClientSOAP\* Arc::SRMClient::client [protected]

SOAP client object.

### 5.45.4.3 SRMImplementation Arc::SRMClient::implementation [protected]

The implementation of the server.

### 5.45.4.4 Logger Arc::SRMClient::logger [static, protected]

Logger.

### 5.45.4.5 NS Arc::SRMClient::ns [protected]

SOAP namespace.

**5.45.4.6** `std::string Arc::SRMClient::service_endpoint` [protected]

URL of the service endpoint, eg `http://srm.host.org:8443/srm/managerv2` All SURIs passed to methods must correspond to this endpoint.

**5.45.4.7** `time_t Arc::SRMClient::user_timeout` [protected]

Timeout for requests to the SRM service.

**5.45.4.8** `std::string Arc::SRMClient::version` [protected]

The version of the SRM protocol used.

The documentation for this class was generated from the following file:

- `SRMClient.h`

## 5.46 Arc::SRMClientRequest Class Reference

Class to represent a SRM request.

```
#include <SRMClientRequest.h>
```

### Public Member Functions

- [SRMClientRequest](#) (const std::list< std::string > &urls) throw (SRMInvalidRequestException)
- [SRMClientRequest](#) (const std::string &url="", const std::string &id="") throw (SRMInvalidRequestException)
- std::string [url](#) () const
- int [waiting\\_time](#) () const
- void [finished\\_success](#) ()
- void [finished\\_partial\\_success](#) ()
- void [finished\\_error](#) ()
- void [finished\\_abort](#) ()
- void [wait](#) (int t=0)
- void [cancelled](#) ()
- SRMRequestStatus [status](#) () const

### 5.46.1 Detailed Description

Class to represent a SRM request.

It may be used for multiple operations, for example calling getTURLs() sets the request token in the request object (for a v2.2 client) and then same object is passed to releaseGet().

### 5.46.2 Constructor & Destructor Documentation

#### 5.46.2.1 Arc::SRMClientRequest::SRMClientRequest (const std::list< std::string > &urls) throw (SRMInvalidRequestException) [inline]

Creates a request object with multiple URLs.

The URLs here are in the form srm://srm.host.org/path/to/file

#### 5.46.2.2 Arc::SRMClientRequest::SRMClientRequest (const std::string &url = "", const std::string &id = "") throw (SRMInvalidRequestException) [inline]

Creates a request object with a single URL.

The URL here is in the form srm://srm.host.org/path/to/file

### 5.46.3 Member Function Documentation

#### 5.46.3.1 void Arc::SRMClientRequest::cancelled () [inline]

Set status to SRM\_REQUEST\_CANCELLED.

**5.46.3.2 void Arc::SRMClientRequest::finished\_abort () [inline]**

Set status to SRM\_REQUEST\_SHOULD\_ABORT.

**5.46.3.3 void Arc::SRMClientRequest::finished\_error () [inline]**

Set status to SRM\_REQUEST\_FINISHED\_ERROR.

**5.46.3.4 void Arc::SRMClientRequest::finished\_partial\_success () [inline]**

Set status to SRM\_REQUEST\_FINISHED\_PARTIAL\_SUCCESS.

**5.46.3.5 void Arc::SRMClientRequest::finished\_success () [inline]**

Set status to SRM\_REQUEST\_FINISHED\_SUCCESS.

**5.46.3.6 SRMRequestStatus Arc::SRMClientRequest::status () const [inline]**

Get status.

**5.46.3.7 std::string Arc::SRMClientRequest::surl () const [inline]**

Returns the first surl in the list.

**5.46.3.8 void Arc::SRMClientRequest::wait (int t = 0) [inline]**

Set waiting time to t and status to SRM\_REQUEST\_ONGOING.

**5.46.3.9 int Arc::SRMClientRequest::waiting\_time () const [inline]**

Get waiting time. A waiting time of zero means no estimate was given by the remote service.

The documentation for this class was generated from the following file:

- SRMClientRequest.h

## 5.47 SRMFileInfo Class Reference

```
#include <SRMInfo.h>
```

### 5.47.1 Detailed Description

Info about a particular entry in the SRM info file

The documentation for this class was generated from the following file:

- SRMInfo.h

## 5.48 Arc::SRMFileMetaData Struct Reference

SRM-related file metadata.

```
#include <SRMClient.h>
```

### 5.48.1 Detailed Description

SRM-related file metadata.

The documentation for this struct was generated from the following file:

- SRMClient.h

## 5.49 SRMInfo Class Reference

```
#include <SRMInfo.h>
```

### 5.49.1 Detailed Description

Represents SRM info stored in file. A combination of host and SRM version make a unique entry.

The documentation for this class was generated from the following file:

- SRMInfo.h

## 5.50 Arc::SRMInvalidRequestException Class Reference

General exception to represent a bad SRM request.

```
#include <SRMClientRequest.h>
```

### 5.50.1 Detailed Description

General exception to represent a bad SRM request.

The documentation for this class was generated from the following file:

- SRMClientRequest.h



## 5.51 ArcSec::UsernameTokenSH Class Reference

Adds WS-Security Username Token into SOAP Header.

```
#include <UsernameTokenSH.h>
```

### 5.51.1 Detailed Description

Adds WS-Security Username Token into SOAP Header.

The documentation for this class was generated from the following file:

- UsernameTokenSH.h

## 5.52 ArcSHCLegacy::voms Struct Reference

```
#include <auth.h>
```

### Data Fields

- std::string [server](#)
- std::string [voname](#)
- std::vector< [voms\\_attrs](#) > [attrs](#)

### 5.52.1 Detailed Description

VOMS data

### 5.52.2 Field Documentation

#### 5.52.2.1 std::vector<[voms\\_attrs](#)> [ArcSHCLegacy::voms::attrs](#)

User's characteristics

#### 5.52.2.2 std::string [ArcSHCLegacy::voms::server](#)

The VOMS server DN, as from its certificate

#### 5.52.2.3 std::string [ArcSHCLegacy::voms::voname](#)

The name of the VO to which the VOMS belongs

The documentation for this struct was generated from the following file:

- [auth.h](#)

## 5.53 ArcSHCLegacy::voms\_attrs Struct Reference

```
#include <auth.h>
```

### Data Fields

- std::string [group](#)
- std::string [role](#)
- std::string [cap](#)

### 5.53.1 Detailed Description

VOMS attributes

### 5.53.2 Field Documentation

#### 5.53.2.1 std::string [ArcSHCLegacy::voms\\_attrs::cap](#)

user's capability

#### 5.53.2.2 std::string [ArcSHCLegacy::voms\\_attrs::group](#)

user's group

#### 5.53.2.3 std::string [ArcSHCLegacy::voms\\_attrs::role](#)

user's role

The documentation for this struct was generated from the following file:

- [auth.h](#)

## 5.54 ArcSec::X509TokenSH Class Reference

Adds WS-Security X509 Token into SOAP Header.

```
#include <X509TokenSH.h>
```

### 5.54.1 Detailed Description

Adds WS-Security X509 Token into SOAP Header.

The documentation for this class was generated from the following file:

- X509TokenSH.h

## 5.55 ArcSec::XACMLAlgFactory Class Reference

Algorithm factory class for XACML.

```
#include <XACMLAlgFactory.h>
```

### Public Member Functions

- virtual CombiningAlg \* [createAlg](#) (const std::string &type)

#### 5.55.1 Detailed Description

Algorithm factory class for XACML.

#### 5.55.2 Member Function Documentation

##### 5.55.2.1 virtual CombiningAlg\* ArcSec::XACMLAlgFactory::createAlg (const std::string & type) [virtual]

return a Alg object according to the "CombiningAlg" attribute in the <Policy> node; The [XACMLAlgFactory](#) itself will release the Alg objects

The documentation for this class was generated from the following file:

- XACMLAlgFactory.h

## 5.56 ArcSec::XACMLAttributeFactory Class Reference

Attribute factory class for XACML specified attributes.

```
#include <XACMLAttributeFactory.h>
```

### Public Member Functions

- virtual AttributeValue \* [createValue](#) (const Arc::XMLNode &node, const std::string &type)

### 5.56.1 Detailed Description

Attribute factory class for XACML specified attributes.

### 5.56.2 Member Function Documentation

#### 5.56.2.1 virtual AttributeValue\* ArcSec::XACMLAttributeFactory::createValue (const Arc::XMLNode & node, const std::string & type) [virtual]

creat a AttributeValue according to the value in the XML node and the type; It should be the caller to release the AttributeValue Object

The documentation for this class was generated from the following file:

- XACMLAttributeFactory.h

## 5.57 ArcSec::XACMLAttributeProxy< TheAttribute > Class Template Reference

XACML specific AttributeProxy class.

```
#include <XACMLAttributeProxy.h>
```

### Public Member Functions

- virtual AttributeValue \* [getAttribute](#) (const Arc::XMLNode &node)

#### 5.57.1 Detailed Description

`template<class TheAttribute> class ArcSec::XACMLAttributeProxy< TheAttribute >`

XACML specific AttributeProxy class.

#### 5.57.2 Member Function Documentation

**5.57.2.1** `template<class TheAttribute> AttributeValue * ArcSec::XACMLAttributeProxy< TheAttribute >::getAttribute (const Arc::XMLNode & node)` [virtual]

Implementation of getAttribute method.

The documentation for this class was generated from the following file:

- XACMLAttributeProxy.h

## 5.58 ArcSec::XACMLCondition Class Reference

[XACMLCondition](#) class to parse and operate XACML specific <Condition> node.

```
#include <XACMLCondition.h>
```

### Public Member Functions

- [XACMLCondition](#) (Arc::XMLNode &node, EvaluatorContext \*ctx)

### 5.58.1 Detailed Description

[XACMLCondition](#) class to parse and operate XACML specific <Condition> node.

### 5.58.2 Constructor & Destructor Documentation

#### 5.58.2.1 ArcSec::XACMLCondition::XACMLCondition (Arc::XMLNode & *node*, EvaluatorContext \* *ctx*)

Constructor -

The documentation for this class was generated from the following file:

- XACMLCondition.h



## 5.59 ArcSec::XACMLEvaluationCtx Class Reference

EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc.

```
#include <XACMLEvaluationCtx.h>
```

### Public Member Functions

- [XACMLEvaluationCtx](#) (Request \*request)

#### 5.59.1 Detailed Description

EvaluationCtx, in charge of storing some context information for evaluation, including Request, current time, etc.

#### 5.59.2 Constructor & Destructor Documentation

##### 5.59.2.1 ArcSec::XACMLEvaluationCtx::XACMLEvaluationCtx (Request \* *request*)

Construct a new EvaluationCtx based on the given request

The documentation for this class was generated from the following file:

- XACMLEvaluationCtx.h

## 5.60 ArcSec::XACMLEvaluator Class Reference

Execute the policy evaluation, based on the request and policy.

```
#include <XACMLEvaluator.h>
```

### Public Member Functions

- virtual Response \* [evaluate](#) (Request \*request)

#### 5.60.1 Detailed Description

Execute the policy evaluation, based on the request and policy.

#### 5.60.2 Member Function Documentation

##### 5.60.2.1 virtual Response\* ArcSec::XACMLEvaluator::evaluate (Request \* *request*) [virtual]

Evaluate the request based on the policy information inside PolicyStore

The documentation for this class was generated from the following file:

- XACMLEvaluator.h

## 5.61 ArcSec::XACMLFnFactory Class Reference

Function factory class for XACML specified attributes.

```
#include <XACMLFnFactory.h>
```

### Public Member Functions

- virtual Function \* [createFn](#) (const std::string &type)

#### 5.61.1 Detailed Description

Function factory class for XACML specified attributes.

#### 5.61.2 Member Function Documentation

**5.61.2.1** virtual Function\* ArcSec::XACMLFnFactory::createFn (const std::string & *type*)  
[virtual]

return a Function object according to the "Function" attribute in the XML node; The [XACMLFnFactory](#) itself will release the Function objects

The documentation for this class was generated from the following file:

- XACMLFnFactory.h

## 5.62 ArcSec::XACMLPDP Class Reference

[XACMLPDP](#) - PDP which can handle the XACML specific request and policy schema.

```
#include <XACMLPDP.h>
```

### 5.62.1 Detailed Description

[XACMLPDP](#) - PDP which can handle the XACML specific request and policy schema.

The documentation for this class was generated from the following file:

- XACMLPDP.h

## 5.63 ArcSec::XACMLPolicy Class Reference

[XACMLPolicy](#) class to parse and operate XACML specific <Policy> node.

```
#include <XACMLPolicy.h>
```

### Public Member Functions

- [XACMLPolicy](#) (Arc::PluginArgument \*parg)
- [XACMLPolicy](#) (const Arc::XMLNode node, Arc::PluginArgument \*parg)
- [XACMLPolicy](#) (const Arc::XMLNode node, EvaluatorContext \*ctx, Arc::PluginArgument \*parg)
- virtual void [make\\_policy](#) ()

### 5.63.1 Detailed Description

[XACMLPolicy](#) class to parse and operate XACML specific <Policy> node.

### 5.63.2 Constructor & Destructor Documentation

#### 5.63.2.1 ArcSec::XACMLPolicy::XACMLPolicy (Arc::PluginArgument \*parg)

Constructor

#### 5.63.2.2 ArcSec::XACMLPolicy::XACMLPolicy (const Arc::XMLNode node, Arc::PluginArgument \*parg)

Constructor

#### 5.63.2.3 ArcSec::XACMLPolicy::XACMLPolicy (const Arc::XMLNode node, EvaluatorContext \*ctx, Arc::PluginArgument \*parg)

Constructor -

### 5.63.3 Member Function Documentation

#### 5.63.3.1 virtual void ArcSec::XACMLPolicy::make\_policy () [virtual]

Parse XMLNode, and construct the low-level Rule object

The documentation for this class was generated from the following file:

- XACMLPolicy.h

## 5.64 ArcSec::XACMLRule Class Reference

[XACMLRule](#) class to parse XACML specific <Rule> node.

```
#include <XACMLRule.h>
```

### 5.64.1 Detailed Description

[XACMLRule](#) class to parse XACML specific <Rule> node.

The documentation for this class was generated from the following file:

- XACMLRule.h

## 5.65 ArcSec::XACMLTarget Class Reference

[XACMLTarget](#) class to parse and operate XACML specific <Target> node.

```
#include <XACMLTarget.h>
```

### Public Member Functions

- [XACMLTarget](#) (Arc::XMLNode &node, EvaluatorContext \*ctx)

### 5.65.1 Detailed Description

[XACMLTarget](#) class to parse and operate XACML specific <Target> node.

### 5.65.2 Constructor & Destructor Documentation

#### 5.65.2.1 ArcSec::XACMLTarget::XACMLTarget (Arc::XMLNode & *node*, EvaluatorContext \* *ctx*)

Constructor -

The documentation for this class was generated from the following file:

- XACMLTarget.h

# Index

- ~LDAPQuery
  - Arc::LDAPQuery, 36
- ~PayloadTLSStream
  - ArcMCCTLS::PayloadTLSStream, 54
- ~SRMClient
  - Arc::SRMClient, 61
- abort
  - Arc::SRMClient, 61
- AndList
  - ArcSec, 10
- Arc::DataPointARC, 24
- Arc::DataPointFile, 25
- Arc::DataPointGFAL, 26
- Arc::DataPointGridFTP, 27
- Arc::DataPointHTTP, 28
- Arc::DataPointLDAP, 29
- Arc::DataPointLFC, 30
- Arc::DataPointRLS, 31
- Arc::DataPointSRM, 32
- Arc::DataPointXrootd, 33
- Arc::LDAPQuery, 36
  - ~LDAPQuery, 36
  - LDAPQuery, 36
  - Query, 36
  - Result, 36
- Arc::SRMClient, 60
  - ~SRMClient, 61
  - abort, 61
  - cfg, 67
  - checkPermissions, 61
  - client, 67
  - copy, 62
  - getInstance, 62
  - getRequestTokens, 62
  - getSpaceTokens, 62
  - getURLs, 63
  - getURLsStatus, 63
  - getVersion, 63
  - implementation, 67
  - info, 63, 64
  - logger, 67
  - mkDir, 64
  - ns, 67
  - ping, 64
  - process, 65
  - putURLs, 65
  - putURLsStatus, 65
  - release, 65
  - releaseGet, 66
  - releasePut, 66
  - remove, 66
  - requestBringOnline, 66
  - requestBringOnlineStatus, 67
  - service\_endpoint, 67
  - SRMClient, 61
  - user\_timeout, 68
  - version, 68
- Arc::SRMClientRequest, 69
- Arc::SRMClientRequest
  - cancelled, 69
  - finished\_abort, 69
  - finished\_error, 70
  - finished\_partial\_success, 70
  - finished\_success, 70
  - SRMClientRequest, 69
  - status, 70
  - surl, 70
  - wait, 70
  - waiting\_time, 70
- Arc::SRMFileMetaData, 72
- Arc::SRMInvalidRequestException, 74
- ArcEvaluationCtx
  - ArcSec::ArcEvaluationCtx, 16
- ArcMCCHTTP::MCC\_HTTP, 37
- ArcMCCHTTP::MCC\_HTTP\_Client, 38
- ArcMCCHTTP::MCC\_HTTP\_Service, 39
- ArcMCCHTTP::PayloadHTTP, 48
- ArcMCCHTTP::PayloadHTTP
  - Attribute, 49
  - Attributes, 49
  - attributes\_, 50
  - Body, 49
  - body\_own\_, 50
  - chunk\_size\_, 50
  - chunked\_, 50
  - code\_, 50
  - end\_, 51
  - Flush, 50
  - get\_body, 50



- length\_, 51
- method\_, 51
- PayloadHTTP, 49
- rbody\_, 51
- read, 50
- read\_header, 50
- readline, 50
- reason\_, 51
- sbody\_, 51
- stream\_, 51
- stream\_own\_, 51
- uri\_, 51
- version\_major\_, 51
- version\_minor\_, 51
- ArcMCCSOAP::MCC\_SOAP, 40
- ArcMCCSOAP::MCC\_SOAP\_Service, 41
- ArcMCCTCP::MCC\_TCP, 42
- ArcMCCTCP::MCC\_TCP\_Client, 43
- ArcMCCTCP::MCC\_TCP\_Service, 44
- ArcMCCTCP::MCC\_TCP\_Service
  - MCC\_TCP\_Service, 44
- ArcMCCTCP::PayloadTCPsocket, 53
- ArcMCCTCP::PayloadTCPsocket
  - PayloadTCPsocket, 53
- ArcMCCTLS::MCC\_TLS, 45
- ArcMCCTLS::MCC\_TLS\_Client, 46
- ArcMCCTLS::MCC\_TLS\_Service, 47
- ArcMCCTLS::PayloadTLSStream, 54
- ArcMCCTLS::PayloadTLSStream
  - ~PayloadTLSStream, 54
  - GetCert, 54
  - GetPeerCert, 54
  - PayloadTLSStream, 54
  - ssl\_, 55
  - STACK\_OF, 54
- ArcPolicy
  - ArcSec::ArcPolicy, 20
- ArcSec, 7
- ArcSec
  - AndList, 10
  - Match, 10
  - OrList, 10
- ArcSec::AllowPDP, 11
- ArcSec::ArcAlgFactory, 12
- ArcSec::ArcAlgFactory
  - createAlg, 12
- ArcSec::ArcAttributeFactory, 13
- ArcSec::ArcAttributeFactory
  - createValue, 13
- ArcSec::ArcAttributeProxy, 14
- ArcSec::ArcAttributeProxy
  - getAttribute, 14
- ArcSec::ArcAuthZ, 15
- ArcSec::ArcAuthZ
  - Handle, 15
  - MakePDPs, 15
- ArcSec::ArcEvaluationCtx, 16
- ArcSec::ArcEvaluationCtx
  - ArcEvaluationCtx, 16
  - split, 16
- ArcSec::ArcEvaluator, 17
- ArcSec::ArcEvaluator
  - evaluate, 17
- ArcSec::ArcFnFactory, 18
- ArcSec::ArcFnFactory
  - createFn, 18
- ArcSec::ArcPDP, 19
- ArcSec::ArcPolicy, 20
- ArcSec::ArcPolicy
  - ArcPolicy, 20
  - make\_policy, 20
- ArcSec::ArcRequestItem, 21
- ArcSec::ArcRequestTuple, 22
- ArcSec::ArcRule, 23
- ArcSec::DelegationPDP, 34
- ArcSec::DenyPDP, 35
- ArcSec::PDPSERVICEInvoker, 56
- ArcSec::SAML2SSO\_AssertionConsumerSH, 57
- ArcSec::SAMLTokenSH, 58
- ArcSec::SimpleListPDP, 59
- ArcSec::UsernameTokenSH, 75
- ArcSec::X509TokenSH, 78
- ArcSec::XACMLAlgFactory, 79
- ArcSec::XACMLAlgFactory
  - createAlg, 79
- ArcSec::XACMLAttributeFactory, 80
- ArcSec::XACMLAttributeFactory
  - createValue, 80
- ArcSec::XACMLAttributeProxy, 81
- ArcSec::XACMLAttributeProxy
  - getAttribute, 81
- ArcSec::XACMLCondition, 82
- ArcSec::XACMLCondition
  - XACMLCondition, 82
- ArcSec::XACMLEvaluationCtx, 83
- ArcSec::XACMLEvaluationCtx
  - XACMLEvaluationCtx, 83
- ArcSec::XACMLEvaluator, 84
- ArcSec::XACMLEvaluator
  - evaluate, 84
- ArcSec::XACMLFnFactory, 85
- ArcSec::XACMLFnFactory
  - createFn, 85
- ArcSec::XACMLPDP, 86
- ArcSec::XACMLPolicy, 87
- ArcSec::XACMLPolicy
  - make\_policy, 87
  - XACMLPolicy, 87

- ArcSec::XACMLRule, 88
- ArcSec::XACMLTarget, 89
- ArcSec::XACMLTarget
  - XACMLTarget, 89
- ArcSHCLegacy::voms, 76
- ArcSHCLegacy::voms
  - attrs, 76
  - server, 76
  - voname, 76
- ArcSHCLegacy::voms\_attrs, 77
- ArcSHCLegacy::voms\_attrs
  - cap, 77
  - group, 77
  - role, 77
- Attribute
  - ArcMCCHTTP::PayloadHTTP, 49
- Attributes
  - ArcMCCHTTP::PayloadHTTP, 49
- attributes\_
  - ArcMCCHTTP::PayloadHTTP, 50
- attrs
  - ArcSHCLegacy::voms, 76
- Body
  - ArcMCCHTTP::PayloadHTTP, 49
- body\_own\_
  - ArcMCCHTTP::PayloadHTTP, 50
- cancelled
  - Arc::SRMClientRequest, 69
- cap
  - ArcSHCLegacy::voms\_attrs, 77
- cfg
  - Arc::SRMClient, 67
- checkPermissions
  - Arc::SRMClient, 61
- chunk\_size\_
  - ArcMCCHTTP::PayloadHTTP, 50
- chunked\_
  - ArcMCCHTTP::PayloadHTTP, 50
- client
  - Arc::SRMClient, 67
- code\_
  - ArcMCCHTTP::PayloadHTTP, 50
- copy
  - Arc::SRMClient, 62
- createAlg
  - ArcSec::ArcAlgFactory, 12
  - ArcSec::XACMLAlgFactory, 79
- createFn
  - ArcSec::ArcFnFactory, 18
  - ArcSec::XACMLFnFactory, 85
- createValue
  - ArcSec::ArcAttributeFactory, 13
  - ArcSec::XACMLAttributeFactory, 80
- end\_
  - ArcMCCHTTP::PayloadHTTP, 51
- evaluate
  - ArcSec::ArcEvaluator, 17
  - ArcSec::XACMLEvaluator, 84
- finished\_abort
  - Arc::SRMClientRequest, 69
- finished\_error
  - Arc::SRMClientRequest, 70
- finished\_partial\_success
  - Arc::SRMClientRequest, 70
- finished\_success
  - Arc::SRMClientRequest, 70
- Flush
  - ArcMCCHTTP::PayloadHTTP, 50
- get\_body
  - ArcMCCHTTP::PayloadHTTP, 50
- getAttribute
  - ArcSec::ArcAttributeProxy, 14
  - ArcSec::XACMLAttributeProxy, 81
- GetCert
  - ArcMCCTLS::PayloadTLSStream, 54
- getInstance
  - Arc::SRMClient, 62
- GetPeerCert
  - ArcMCCTLS::PayloadTLSStream, 54
- getRequestTokens
  - Arc::SRMClient, 62
- getSpaceTokens
  - Arc::SRMClient, 62
- getTURLs
  - Arc::SRMClient, 63
- getTURLsStatus
  - Arc::SRMClient, 63
- getVersion
  - Arc::SRMClient, 63
- group
  - ArcSHCLegacy::voms\_attrs, 77
- Handle
  - ArcSec::ArcAuthZ, 15
- implementation
  - Arc::SRMClient, 67
- info
  - Arc::SRMClient, 63, 64
- LDAPQuery
  - Arc::LDAPQuery, 36
- length\_
  - ArcMCCHTTP::PayloadHTTP, 51

- logger
  - Arc::SRMClient, 67
- make\_policy
  - ArcSec::ArcPolicy, 20
  - ArcSec::XACMLPolicy, 87
- MakePDPs
  - ArcSec::ArcAuthZ, 15
- Match
  - ArcSec, 10
- MCC\_TCP\_Service
  - ArcMCCTCP::MCC\_TCP\_Service, 44
- method\_
  - ArcMCCHTTP::PayloadHTTP, 51
- mkDir
  - Arc::SRMClient, 64
- ns
  - Arc::SRMClient, 67
- OrList
  - ArcSec, 10
- PayloadHTTP
  - ArcMCCHTTP::PayloadHTTP, 49
- PayloadTCPSocket
  - ArcMCCTCP::PayloadTCPSocket, 53
- PayloadTLSSStream
  - ArcMCCTLS::PayloadTLSSStream, 54
- ping
  - Arc::SRMClient, 64
- process
  - Arc::SRMClient, 65
- putURLs
  - Arc::SRMClient, 65
- putURLsStatus
  - Arc::SRMClient, 65
- Query
  - Arc::LDAPQuery, 36
- rbody\_
  - ArcMCCHTTP::PayloadHTTP, 51
- read
  - ArcMCCHTTP::PayloadHTTP, 50
- read\_header
  - ArcMCCHTTP::PayloadHTTP, 50
- readline
  - ArcMCCHTTP::PayloadHTTP, 50
- reason\_
  - ArcMCCHTTP::PayloadHTTP, 51
- release
  - Arc::SRMClient, 65
- releaseGet
  - Arc::SRMClient, 66
- releasePut
  - Arc::SRMClient, 66
- remove
  - Arc::SRMClient, 66
- requestBringOnline
  - Arc::SRMClient, 66
- requestBringOnlineStatus
  - Arc::SRMClient, 67
- Result
  - Arc::LDAPQuery, 36
- role
  - ArcSHCLegacy::voms\_attrs, 77
- sbody\_
  - ArcMCCHTTP::PayloadHTTP, 51
- server
  - ArcSHCLegacy::voms, 76
- service\_endpoint
  - Arc::SRMClient, 67
- split
  - ArcSec::ArcEvaluationCtx, 16
- SRMClient
  - Arc::SRMClient, 61
- SRMClientRequest
  - Arc::SRMClientRequest, 69
- SRMFileInfo, 71
- SRMInfo, 73
- ssl\_
  - ArcMCCTLS::PayloadTLSSStream, 55
- STACK\_OF
  - ArcMCCTLS::PayloadTLSSStream, 54
- status
  - Arc::SRMClientRequest, 70
- stream\_
  - ArcMCCHTTP::PayloadHTTP, 51
- stream\_own\_
  - ArcMCCHTTP::PayloadHTTP, 51
- surl
  - Arc::SRMClientRequest, 70
- uri\_
  - ArcMCCHTTP::PayloadHTTP, 51
- user\_timeout
  - Arc::SRMClient, 68
- version
  - Arc::SRMClient, 68
- version\_major\_
  - ArcMCCHTTP::PayloadHTTP, 51
- version\_minor\_
  - ArcMCCHTTP::PayloadHTTP, 51
- voname
  - ArcSHCLegacy::voms, 76
- wait

- Arc::SRMClientRequest, [70](#)
- waiting\_time
  - Arc::SRMClientRequest, [70](#)
- XACMLCondition
  - ArcSec::XACMLCondition, [82](#)
- XACMLEvaluationCtx
  - ArcSec::XACMLEvaluationCtx, [83](#)
- XACMLPolicy
  - ArcSec::XACMLPolicy, [87](#)
- XACMLTarget
  - ArcSec::XACMLTarget, [89](#)